ConnectOpen – Automatic Integration of IoT Devices

Nuria Pazos, Michael Müller, Marco Aeberli, Nabil Ouerhani
Applied Science University Western Switzerland (HES-SO), HE-Arc Engineering School
Switzerland
Nuria.Pazos@he-arc.ch

Abstract— **There exists, today, a wide consensus that Internet of Things (IoT) is creating a wide range of business opportunities for various industries and sectors like Manufacturing, Healthcare, Public infrastructure management, Telecommunications and many others. On the other hand, the technological evolution of IoT facing serious challenges. The fragmentation in terms of communication protocols and data formats at device level is one of these challenges. Vendor specific application architectures, proprietary communication protocols and lack of IoT standards are some reasons behind the IoT fragmentation. In this paper we propose a software enabled framework to address the fragmentation challenge. The framework is based on flexible communication agents that are deployed on a gateway and can be adapted to various devices communicating different data formats using different communication protocol. The communication agent is automatically generated based on specifications and automatically deployed on the Gateway in order to connect the devices to a central platform where data are consolidated and exposed via REST APIs to third party services. Security and scalability aspects are also addressed in this work.**

Keywords—IoT; Gateway; End Device; OSGi, Kura, Communication Agent, MQTT

I. INTRODUCTION

Nowadays it is becoming reality what Vint Cerf predicted some years ago: "*... many appliances around the house, in the office, in the car, on our persons, in the buildings that we work and live in will be instrumented and will be part of the net ... When those appliances are Internet-enabled ... you open up an opportunity for new businesses to manage those devices*."

The enabling key for the recent explosion of the IoT business does not rely on technological advances at the level of electronics or cloud technologies, but on the mastering of the complete enabling technologies, from the sensors/actuators layer to the cloud paradigm, going through the smart communicating systems and the access gateways.

Nevertheless, although the enabling technologies make the IoT concept feasible, a large research effort is still required. Specially, the actual lack of standardization and normalization makes the interoperability among the different building blocks of an IoT system an arduous task.

A. Objectives

The main objective pursues by the current project is the design and implementation of a secure and flexible platform capable of easily connect communicating devices and to integrate domain specific applications to exploit the acquired data.

B. Novelties

The key characteristics offered by the ConnectOpen platform are:

- **Customization**: The platform is able to adapt itself to the client's domain specific needs thanks to an easy integration of the domain specific application.
- **Security**: The platform enables an advanced management of user roles and access permissions. The communication flow's security between the end devices and the central platform is assured.
- **Flexibility**: The integration of new communication devices is highly automated thanks to the concept of communication agents. The integration complexity due to the diversity of communication protocols is managed by the platform itself.
- **Performance** and **scalability**: The architectural choices of the platform ConnectOpen pursue to guarantee the performance and the scalability of the associated devices.

C. Structure of the Paper

This paper is structured as follows: Section II summarizes and compares the related work in order to highlight the advantages introduced by ConnectOpen. The proposed platform architecture is then shown in Section III, while Section IV gives the implementation details for each of the target components. Different use cases are then presented in Section V followed by Section VI which gives the conclusions and sets the perspectives.

II. STATE OF THE ART

While everybody agrees that the main goal of an IoT platform is to help deliver innovations to market faster by reducing solution complexity and offering repeatable foundation for how devices will connect and deliver trusted data to the cloud (or to a central server), the way the existing solutions approach this challenge usually differs.

The majority of surveyed platforms [1] differentiate between data generators and consumers, offering separate APIs for interacting with the end devices from one side, and the served applications from the other. The system architecture is similar to the one depicted in Figure 1 [2].

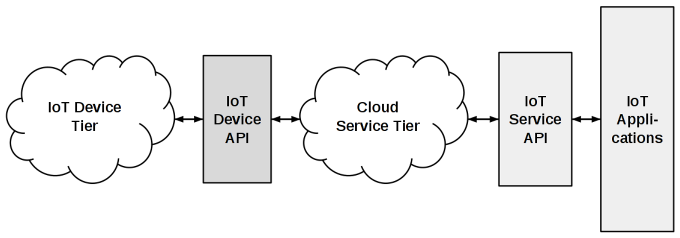

Figure 1. General architecture of a Cloud centric IoT platform

One of the core tasks of every IoT platform is to provide uniform access to the gathered, processed and stored sensing data. For this purpose, the key features to characterize the application-side platform access of the IoT platforms are: The protocols to interact with the platforms, the encoding format, and the interaction pattern. Based on these features, a classification of the existing IoT platforms, such as Xively [4], Etherios [5], Sensei [6], FI-WARE [7], and Osiot [8] is provided in [3].

On the other hand, the access to the end devices, usually implemented through a gateway to mediate and integrate devices that lack native IP connectivity, has been less characterized. Certain IoT platforms restrict the kind and number of attached objects to pre-integrated solutions, which faces scalability hurdles. On the other side, the so-called open IoT solutions, pretend to connect any kind of hardware end device, leaving the development and connection efforts to the third parties such as the providers of communicating objects or the system integrators.

Contrary to the existing approaches, we believe that the data upload process between the communicating node and the storage and visualization platform can be highly automated thanks to an abstraction layer at the gateway level based on the concept of communication agents. The proposed gateway allows thus a seamless integration of several technologies and communications solutions to the ConnectOpen platform.

III. PLATFORM ARCHITECTURE

As introduced before, the ConnectOpen platform is built around a set of building blocks for each layer of the technological stack. Figure 2 shows the different components of the platform, from the sensors/actuators layer to the application layer.

Although the novelty introduced by the proposed IoT solution resides in the gateway concept, it is necessary to introduce the different components of the platform in order to demonstrate the seamless integration of new hardware devices from end-to-end.

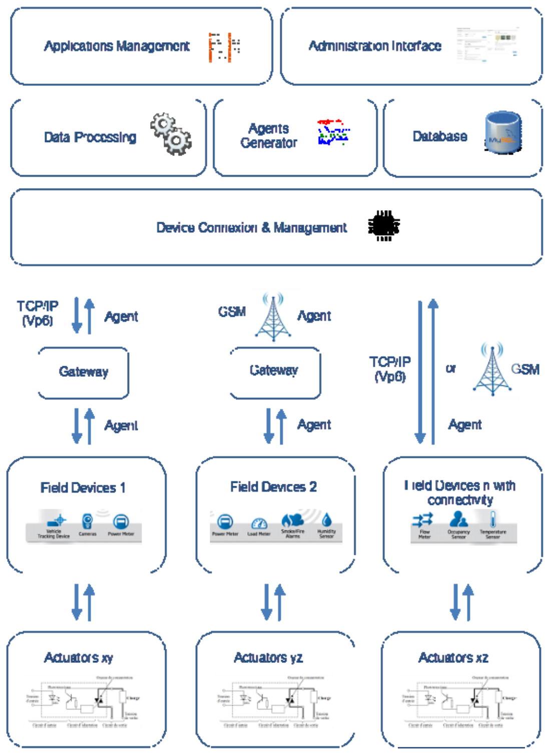

Figure 2. ConnectOpen Architecture

A. *Communicating Devices*

The sensors and actuators are integrated on a single device (embedded system) able to forward the measured values and to receive the commands.

B. *Gateway*

An ubiquitous processing unit capable of communicating with the end devices and, on one side, to push the measured values to the central platform and, on the other side, pull the commands to the end devices.

C. *Broker*

A communication bus capable of forwarding the data between the gateway and the central platform. It is based on the «publish/subscribe» principle thanks to which it assures a high performance and scalability.

D. *Central Platform*

Group of services executed either on the Cloud or on a dedicated server and capable of processing and exploiting the acquired data. The domain specific applications are integrated at this level.

IV. Target Implementation

The entire ConnectOpen platform implements the complete technological stack, from the end device layer (sensor/actuator) to the storage and exploitation layer. At each level, an appropriate technology has been applied for the implementation (see Figure 3): An embedded software development environment for the programming of the end devices, a Java VM for the gateway implementation on a general purpose embedded system, a non-SQL data base (Cassandra) for the data layer, and a web technology for the integration services (REST-API, configuration services, command services, incoming data services and security services) as well as for the business application layer.

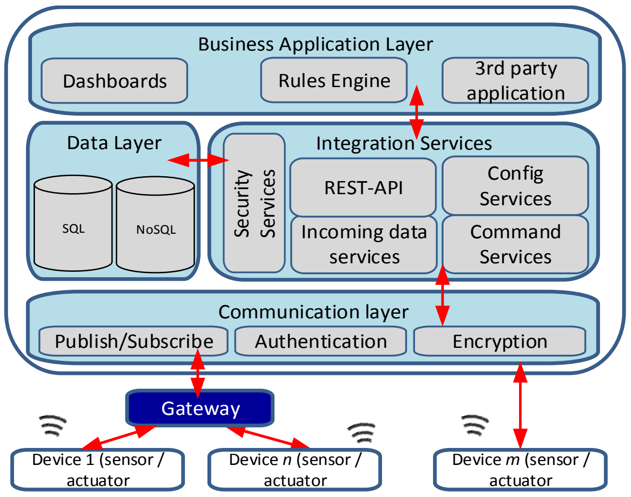

Figure 3. ConnectOpen Implementation

As stated at the introduction, the main goal pursued by the current work is an automatic connection of communicating devices and an easy integration of domain specific applications that exploit the acquired data. To achieve this objective, a new concept of gateway has been introduced which is based on the so-called communication agents. These communication agents allow the seamless integration of communicating devices. Figure 4 shows the different layers of the gateway implementation, where we can identify the hardware platform on the bottom (e.g., a Raspberry Pi) and the communication agents on the top.

On top of the Java VM, the framework OSGi (Open Services Gateway initiative, [9]) has been installed, which is about a modular (built up of bundles) and SOA (Service Oriented Architecture) architecture for deploying and executing Java services on top of a resources-constrained embedded system [10].

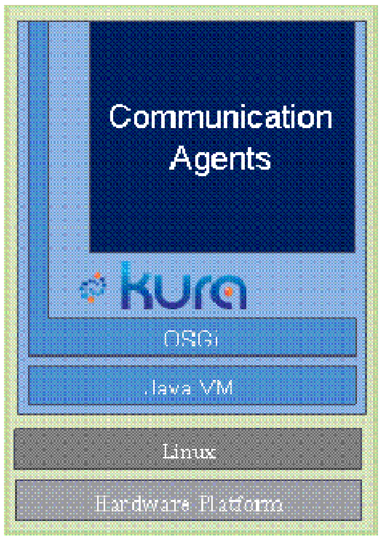

Figure 4. Gateway Implementation

Furthermore, the basic IoT services proposed by Kura [11] (see Figure 5) have been integrated on the implemented architecture on top of the OSGi framework.

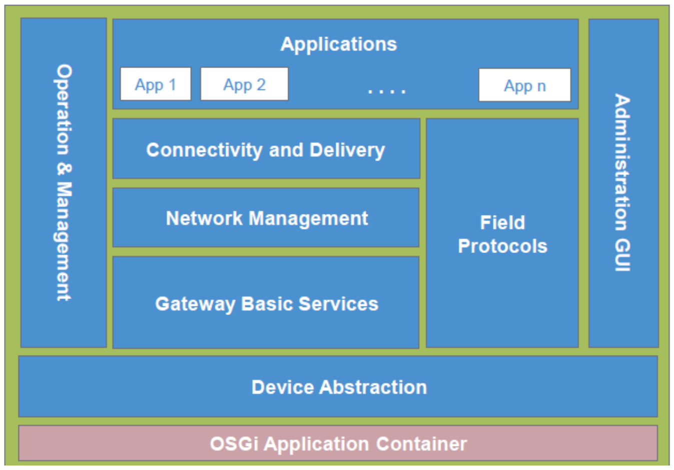

Figure 5. Kura – IoT Services Framework

Some of the IoT services provided by Kura are the following:

- Device Abstraction: Serial, USB, Bluetooth, etc.
- Gateway Basic Services: Watchdog, Embedded DB, Clock Services, etc.
- Network Management: Ethernet, WiFi, etc.
- Connectivity and Delivery: Cloud Service, Data Service, MQTT (Paho), etc.

- Field Protocols: ModBUS, CanBUS, ProfiBUS, etc.
- Operation & Management: Remote configuration, remote update, log service, etc.

Finally, thanks to modularity offered by the framework OSGi, three bundles have been developed which make up the pursued communication agent. Each of the three independent bundles of the OSGi container handles one specific task require to recognize and connect to the end device registered by the IoT application, obtain/send the target data from/to the end device, and publish/subscribe the target data through the MQTT broker to/from the central platform. This architecture as well as the connections between the three bundles which make up the communication agent is depicted in Figure 6.

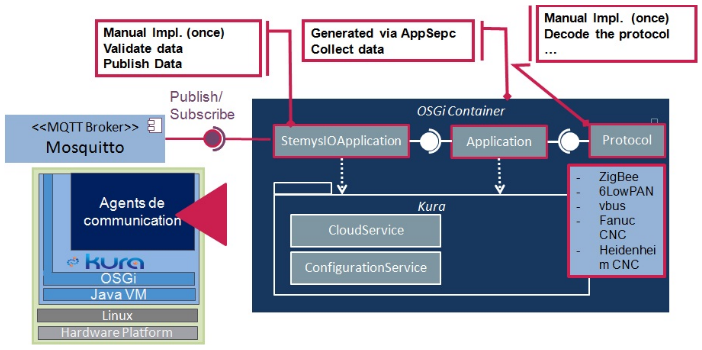

Figure 6. General Architecture of the Communication Agent

One of the key features of the communication agent is the easiness of generation which provides a high flexibility to the platform to integrate new end devices. As shown in Figure 7, the communication agent is implemented from a simple specification of the application key values, such as the version, the metrics to be read, the commands to be written, etc.

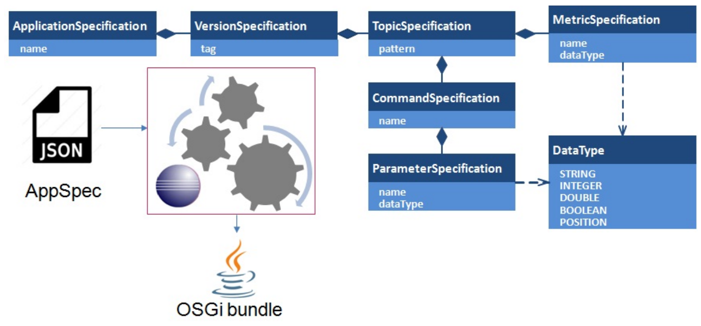

Figure 7. Flow chart for the Implementation of a Communication Agent

V. Use Cases

The key features of the ConnectOpen platform stated in Section I have been validated through the implementation of different use cases directly related to real business applications.

The three scenarios presented below differentiate themselves on the hardware end device (embedded systems built around different vendor specific processing units and sensors), the communication technology (wire/wireless as well as different radio technologies) and the communication topology (star- versus a mesh- topology). Although their diversity, all of them have been able to automatically integrate the ConnectOpen platform by applying the same gateway architecture thanks to the automatic generation of the custom communication agent by means of the application specification.

A. *Intelligent Street Lighting System*

This IoT application targets the adaptation of the illuminating power to the real needs coming from different sources (see Figure 8)

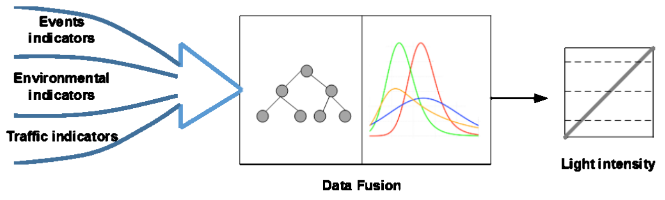

Figure 8. Context-sensitive Street Lighting

The main building blocks of this system are:

1. Sensors: They are about real and virtual sensors (combination of multiple sources of information into a composite data source) able to sense environmental indicators like luminosity, weather forecast, etc. as well as activity indicators like frequency of pedestrians and/or traffic.
2. Central platform: Analyses the data pushed by the sensors to characterize the context and to determine the needs on illuminating power (intense/low activity, good/bad visibility,).
3. Street Lights: Receive the command to regulate the illuminating power according to the determined needs coming from the sensor fusion.

Each street light periodically push/receive data to/from the ConnectOpen platform through the proposed universal gateway. The direct communication link between the gateway and the multiple end devices (sensors and street lights) is established by using an 802.15.4 RF link. The communication agent for this application, which has been automatically generated from the application specification, identifies the communication technology used and is able to parse the data payload encapsulated in the communication protocol.

The main drawback of this configuration is that the distance between the gateway and each of the end devices is limited by the range of the communication technology. If a broader range is required, a mesh topology among the different nodes and towards the gateway can be instead chosen with no structural changes in the gateway architecture required.

B. Smart Parking System

It is about an integrated solution to assist auto drivers to identify free parking places and to reserve them. An assisted navigation system is further proposed to the driver towards the reserved place (see Figure 9).

The key functionalities proposed by this application are: (i) Identification of free parking places; (ii) Reservation of the selected parking place; and (iii) Navigation system to guide the driver.

Figure 9. Smart Park Application

This application is built around the following building blocks:

- Sensors: The magnetic sensors deployed in the parking places allow to pushing the occupancy information to the central platform.
- Central platform: It is mainly used to centralize the sensing data to manage the parking places (status, reservations, etc.). It also provides a REST API to allow tier applications (such as a mobile application) to exploit the sensing data stored in it.
- Mobile application: This application is intended to the auto drivers who would like to identify and reserve a free parking place and to be further guided to it.

For the implementation of this real use case, a mesh topology has been applied to connect the magnetic sensors to the proposed universal gateway and thus enabling a larger range. A 6LoWPAN protocol stack has been applied to route the IPv6 packets containing the sensing data to the gateway and further to the central platform. Since the sensor network is kept local and thus not directly addressable from Internet, the communication agent implements the functionality of querying the border router about the reachability of the different sensing nodes and the path to reach them. Figure 10 shows the software architecture of the Border Router (communicating node which creates the 6LoWPAN network thanks to the RPL routing protocol) and of each End Device, which are placed at each parking place and push the free/occupy information to the central platform through the Border Router directly attached to the gateway. Both types of nodes use the uIP TCP/IP stack [12] implementation for the 6LoWPAN communication.

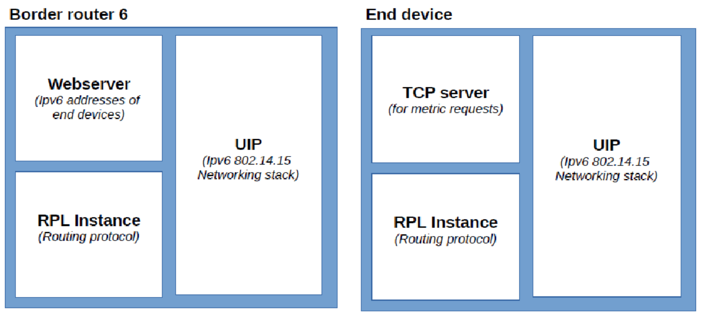

Figure 10. Implementation of Border Router and End Devices in a 6LoWPAN Network

C. Satellite Tracking Device

The main objective of this third example is to demonstrate the applicability of the platform concept introduced by ConnectOpen to interface a sensing node that communicates directly (without a physical gateway) to the central platform. For this purpose, a virtual gateway is implemented on the cloud (or on a central server) which logically corresponds to the physical gateway implemented on an embedded system, such as a Raspberry Pi (use cases A. et B.)

The satellite tracking application (see Figure 11) consists of a geo-tracking device provided of a GPS antenna, for implementing the localization system, and a GSM antenna, for the GPRS communication. One of the main applications of such communicating node is fleet management and real-time tracking and monitoring.

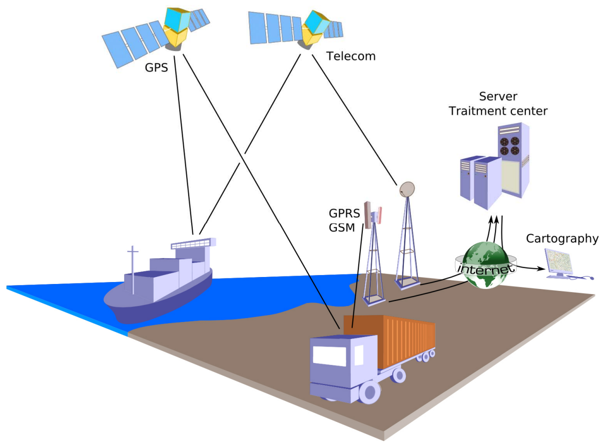

Figure 11. Satellite Tracking Application [13]

VI. Conclusion and Future Work

This paper presented a complete Internet of Things (IoT) platform that enhances the state of the art IoT platforms and framework with two main contributions. Regarding the first contribution, the presented platform addresses the challenge of IoT fragmentation in terms of communication protocols and application data formats. Enhancing existing open source IoT frameworks, the presented work introduced the concept of flexible, configurable and modular communication agents, which clearly separates the three concerns at Gateway level, namely: a) protocol interpretation, b) data parsing, and c) data push to a central platform, each of which is implemented as an OSGi bundle. The advantage of this modular architecture is the speed up of device connection, which is a first step towards plug and play architecture. The second contribution of the presented work is the automatic generation of communication agents using application specifications. We developed an eclipse plug-in that generates OSGi bundles implementing the three modules of the communication agent (protocol bundle, parsing bundle, push bundle). As the push bundle is mostly the same for all device applications and the protocol bundle is highly reusable for devices using the same communication protocol, the major effort has been put in the automatic generation of parsing bundle. The second contribution contributes also to mid-term objective of our work, namely a plug and play IoT platform by providing a corner stone of a Model Driven Architecture of gateways and the associated communication agents.

Acknowledgment

This research is the result of the project ConnectOpen, funded by the Swiss Commission for Technology and Innovation, CTI.

References

[1] T. Menzel, N. Karowski, D. Happ, V. Handziski, and A. Wolisz. “Social Sensor Cloud: An Architecture Meeting Cloud-centric IoT Platform Requirements” Telecommunications Networks Group, TU Berlin, April 2014.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. “Internet of Things (IoT): A vision, architecctural elements, and future directions” Future Generation Computer Systems, 2013.

[3] T. Menzel, N. Karowski, D. Happ, V. Handziski, and A. Wolisz.. “Survey of Cloud-centric IoT Platforms”, 2013.

[4] Xively. [Online]. Available: http://xively.com

[5] Etherios. [Online]. Available: http://www.etherios.com

[6] Sensei. [Online]. Available: http://sensei-project.eu

[7] FI-WARE. [Online]. Available : http://fi-ware.eu

[8] Osiot. [Online]. Available : http://osiot.org

[9] OSGi Alliance [Online]. Available : http://www.osgi.org

[10] R. Alcarria, T. Robles, A. Morales Dominguez, and S. Gonzalez-Miranda. “Flexible Service Composition Based on Bundle Communication in OSGi”. KSII Transactions on Internet and Information Systems 6(1): 116-130, 2012.

[11] Kura [Online]. Available : https://eclipse.org/kura/

[12] « The uIP Embedded TCP/IP Stack » ; The uIP 1.0 Reference Manual

[13] «Geolocation» par Éric Chassaing — Travail personnel. Sous licence CC BY-SA 3.0 via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Geolocation.png#/media/File:Geolocation